# Online Action Language $o\mathcal{BC}+$

Joseph Babb  Joohyung Lee

Automated Reasoning Group
Arizona State University, USA

LPNMR 2015

# Motivation

- While Answer Set Programming (ASP) is being widely applied to many challenging problems, most ASP applications are limited to offline usages.
- There is a need for online answer set solving in many real-time dynamic systems : robotics, policies, sensors, stream reasoning. ("How do I call ASP solvers?")
- Continuous grounding and solving in view of possible yet unknown future events is one of the main challenges in applying ASP to real-time dynamic systems.

# Online Answer Set Programming [Gebser *et al.*, LPNMR'11]

- Incrementally ground and compose program slices taking into account external knowledge acquired asynchronously.
  - Thereby avoiding multiple unnecessary restarts of the grounding/solving process for each arrival of external inputs.
- OCLINGO is an online ASP solver that processes external inputs provided at runtime by a controller.
- However, checking the syntactic requirement for sound execution of online answer set solving is quite a complex task for the user, which significantly limits the usability of online answer set programming.

# Online Action Language $\mathcal{BC}+$

- We address this challenge by introducing an online extension of high level action language $\mathcal{BC}+$, which we call $o\mathcal{BC}+$.
- $o\mathcal{BC}+$ is implemented in CPLUS2ASP. In addition to the static and the incremental mode already available, newly introduced is the reactive mode, which invokes OCLINGO v3.0.92.
  - (Cannot work with CLINGO v4.X yet due to the incompatibility of the input languages in the component software F2LP: `#domain` directive was gone)
  - Another high level language for OCLINGO ensuring the modularity condition of OCLINGO is Online Agent Logic Programming language from [Cerexhe, Gebser & Thielscher, 2014]. However, Agent Logic Programs are less expressive than action languages (e.g., no negation).

# Some Examples in $o\mathcal{BC}+$ (Tested on CPLUS2ASP)

- Tower of Hanoi: an antagonistic entity is able to move pegs in order to attempt to thwart the agent's efforts.
- Blocks World: An external antagonist has the ability to 'rain' more blocks on the table at any given time.
- Warehouse Management: The agent is set to controlling multiple robots within an automated warehouse. The agent must plan the path of each robot in order to retrieve items requested by external customers while simultaneously avoiding collisions.
- Online Scheduling: The agent is placed in charge of a number of machines specialized to perform a specific set of jobs. The agent must allocate the resources provided by each machine in order to finish each request job. Meanwhile, machines may fail or be repaired.
- Elevator Control: The agent is to service a number of elevator requests made by individuals.

# (Offline) Action Language $\mathcal{BC}+$

- Successor of $\mathcal{BC}$ [Lee, Lifschitz and Yang, IJCAI 2013]. Generalizes both $\mathcal{B}$ and $\mathcal{C}+$.
- The main idea is to define the semantics of $\mathcal{BC}+$ in terms of propositional formulas under the stable model semantics [Ferraris, 2005].
- Modern ASP language constructs, such as choice rules and aggregates, can be viewed as an abbreviation of formulas under the stable model semantics.
- Enhancements in ASP are readily applied in the setting of action languages: online answer set solving.

# Syntax of $\mathcal{BC}+$

We consider propositional formulas whose signature consists of atoms of the form $c = v$, where $c$ is called a constant and is associated with a finite set called the domain. Constants are either fluents or actions.

$\{c = v\}^{\mathrm{ch}}$ stands for $(c = v) \vee \neg(c = v)$.
Intuitive reading: "by default, $c$ has the value $v$."

- Static law: **caused** $F$ **if** $G$ ($F$, $G$ are fluent formulas)

"The light is usually on while the switch is on":
**caused** $\{Light = On\}^{\mathrm{ch}}$ **if** $Switch = On$
Alternatively: **default** $Light = On$ **if** $Switch = On$

# Syntax of $\mathcal{BC}+$

We consider propositional formulas whose signature consists of atoms of the form $c = v$, where $c$ is called a constant and is associated with a finite set called the domain. Constants are either fluents or actions.

$\{c = v\}^{\mathrm{ch}}$ stands for $(c = v) \vee \neg(c = v)$.
Intuitive reading: "by default, $c$ has the value $v$."

- Static law: **caused** $F$ **if** $G$ ($F$, $G$ are fluent formulas)

"The light is usually on while the switch is on":
**caused** $\{Light = On\}^{\mathrm{ch}}$ **if** $Switch = On$
Alternatively: **default** $Light = On$ **if** $Switch = On$

- Action dynamic law: **caused** $F$ **if** $G$ ($F$ is an action formula)

"The agent may move to arbitrary locations":
**caused** $\{Move = l\}^{\mathrm{ch}}$ **if** $\top$ (for all $l \in Locations$)
Alternatively: **exogenous** $Move$

# Syntax of $\mathcal{BC}+$

- Fluent dynamic law: **caused** $F$ **if** $G$ **after** $H$ ($F$, $G$: fluent formulas)

  The effect of *Move*:

  **caused** $Loc = l$ **if** $\top$ **after** $Move = \mathbf{t}$

  Alternatively: *Move* **causes** $Loc = l$

  "The agent's location is inertial":

  **caused** $\{Loc = l\}^{\mathrm{ch}}$ **if** $\top$ **after** $Loc = l$ (for all $l \in Locations$)

  Alternatively:

  **inertial** *Loc*

# Semantics

- The semantics of $\mathcal{BC}+$ description $D$ is given in terms of a sequence of propositional formulas $PF_0(D), PF_1(D), \ldots$ under the stable model semantics.

| $D$ | $PF_m(D)$ |
|---|---|
| **caused** $F$ **if** $G$ | $i:F \leftarrow i:G$ $\quad (i = 0, 1, \ldots)$ |
| **caused** $F$ **if** $G$ **after** $H$ | $i+1:F \leftarrow (i+1:G) \wedge (i:H)$ $\quad (i = 0, 1, \ldots)$ |
| | $\{0:c=v\}^{\mathrm{ch}}$ for every regular fluent $c$ and every $v$ |
| | Uniq. and exist. of value constraints for $i:c$ |

- The states of the transition system described by $D$ correspond to the stable models of $PF_0(D)$.
- The transitions correspond to the stable models of $PF_1(D)$.

**Theorem** The stable models of $PF_m(D)$ are in a 1-1 correspondence with the paths of length $m$ in the transition system $D$.

# Online $\mathcal{BC}+$

The signature hierarchy of $o\mathcal{BC}+$ is extended from that of (offline) $\mathcal{BC}+$.

<table>
<tr><td colspan="4">signature $\sigma$</td></tr>
<tr><td colspan="3">fluents ($\sigma^{fl}$)</td><td>actions ($\sigma^{act}$)</td></tr>
<tr><td colspan="2">internal</td><td rowspan="2">external ($\sigma^{ef}$)</td><td rowspan="2">internal</td><td rowspan="2">external ($\sigma^{ea}$)</td></tr>
<tr><td>regular</td><td>static. det.</td></tr>
</table>

The domain of each external fluent and action constant contains a special element $\mathtt{u}$, which represents an *unknown* value.

The syntax of causal laws is defined the same as in $\mathcal{BC}+$ except that external constants are allowed in the bodies but not in the heads.

# Observation / Observational Constraint

An observation is an expression of the form

$$\textbf{observed } c = v \textbf{ at } m \tag{1}$$

where $c$ is an external constant, $v$ is a value other than $\mathtt{u}$, and $m$ is a nonnegative integer.

An observational constraint is an expression of the form

$$\textbf{constraint } F \textbf{ at } m \tag{2}$$

where $F$ is a propositional formula containing no external constants and $m$ is a nonnegative integer.

An observation stream is a set of observations and observational constraints.

# Faulty Switch in $o\mathcal{BC}+$

**inertial** *Sw*
**exogenous** *Flip*

*Flip* **causes** *Sw* = `on` **if** *Sw* = `off`
*Flip* **causes** *Sw* = `off` **if** *Sw* = `on`
**default** *Light* = *s* **if** *Sw* = *s*

**inertial** *Fault* **after** *ReplaceBulb* = `f`
**exogenous** *ReplaceBulb*

**nonexecutable** *ReplaceBulb* **if** *Flip* = `t`
**caused** *Fault* = *v* **if** *ExtFault* = *v*
**caused** *Light* = `off` **if** *Fault* = `t`
**default** *Fault* = `u` **after** *ReplaceBulb*

# Normal History

It is more meaningful to assume that the external input is "abnormal" to the system dynamics, and we want to "minimize" their effects.

Intuitively, in a normal history, the external constants are mapped to an unknown value unless the external observation asserts otherwise.

We say that history $\mathcal{H}$ observes an observation stream $\mathcal{O}$ if,

- for each observation **observed** $c = v$ **at** $m$ in $\mathcal{O}$, history $\mathcal{H}$ satisfies $m : c = v$, and
- for each observational constraint **constraint** $F$ **at** $m$ in $\mathcal{O}$, history $\mathcal{H}$ satisfies $m : F$.

We say that $\mathcal{H}$ is normal with respect to $\mathcal{O}$, if it observes $\mathcal{O}$, and, for each external constant $c$ and each $i$, $\mathcal{H}$ satisfies $i : c = \mathrm{u}$ when there is no observation in $\mathcal{O}$ telling the value of $c$ at $i$.

# Example: Faulty Switch

The minimum length history from

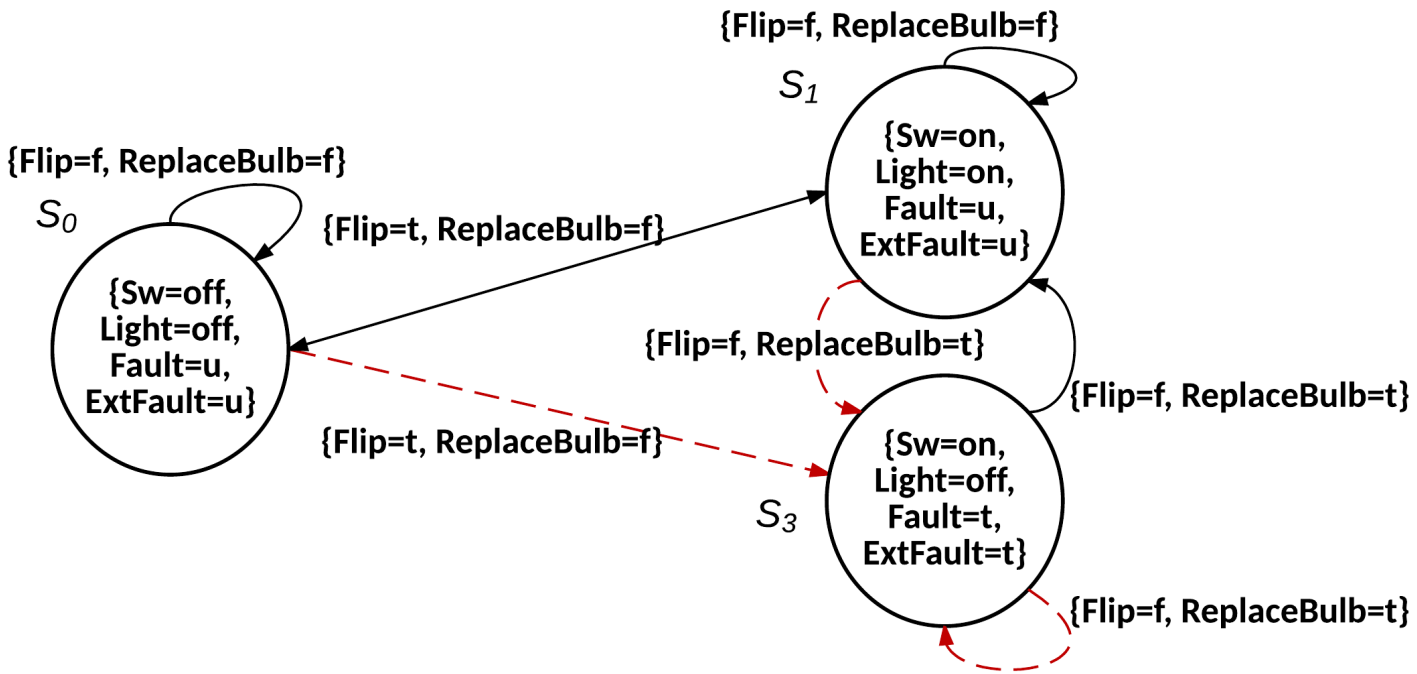
$$\mathcal{S}_0 = \{\mathit{Switch} = \mathtt{off},\ \mathit{Light} = \mathtt{off},\ \mathit{Fault} = \mathtt{u},\ \mathit{ExtFault} = \mathtt{u}\}$$

to a state $\mathcal{S}$ such that $\mathcal{S} \models \mathit{Light} = \mathtt{on}$ are $\langle \mathcal{S}_0, \mathcal{E}_0, \mathcal{S}_1 \rangle$, and $\langle \mathcal{S}_0, \mathcal{E}_0, \mathcal{S}_2 \rangle$ where

$$\begin{aligned}
\mathcal{E}_0 &= \{\mathit{Flip} = \mathtt{t},\ \mathit{ReplaceBulb} = \mathtt{f}\}, \\
\mathcal{S}_1 &= \{\mathit{Switch} = \mathtt{on},\ \mathit{Light} = \mathtt{on},\ \mathit{Fault} = \mathtt{u},\ \mathit{ExtFault} = \mathtt{u}\}, \text{and} \\
\mathcal{S}_2 &= \{\mathit{Switch} = \mathtt{on},\ \mathit{Light} = \mathtt{on},\ \mathit{Fault} = \mathtt{f},\ \mathit{ExtFault} = \mathtt{f}\}.
\end{aligned}$$

Of the two, only $\langle \mathcal{S}_0, \mathcal{E}_0, \mathcal{S}_1 \rangle$ is normal with respect to the online progression $\mathcal{O} = [\ ]$.

# Example: Faulty Switch




If, following the execution of *Flip*, the agent observes that a fault did occur:

$$\mathcal{O} = [\{\textbf{observed } \mathit{ExtFault} = \mathtt{t} \textbf{ at } 1,\ \textbf{constraint } \mathit{Flip} = \mathtt{t} \textbf{ at } 0\}].$$

The new minimum length history from $\mathcal{S}_0$ to a state $\mathcal{S}$ such that $\mathcal{S} \models \mathit{Light} = \mathtt{on}$ and is normal w.r.t. $\mathcal{O}$ is $\langle \mathcal{S}_0, \mathcal{E}_0, \mathcal{S}_3, \mathcal{E}_1, \mathcal{S}_1 \rangle$.

# Incremental Composition with Online Progression

The semantics of $o\mathcal{BC}+$ is defined by translation into incremental theory and online progression, which is extended from [Gebser *et al.*, 2011] to propositional formulas.

An incremental theory is a triple $\langle B, P[t], Q[t] \rangle$, where

- $B$ is the base component.
- $P[t]$ is the cumulative component.
- $Q[t]$ is the volatile component.

An online progression $\langle E_i[e_i], F_i[f_i] \rangle_j$ is a sequence of pairs of formulas $(E_i[e_i], F_i[f_i])$ for $i = 1, \ldots, j$, each $E_i[e_i]$ and $F_i[f_i]$ corresponds to stable and volatile knowledge acquired during execution, respectively.
(For example, $E_4[3]$ is the fourth piece of online input and contains information relevant to step 3.)

# (Simple) Expansion vs. Incremental Composition

Given an incremental theory $\langle B, P[t], Q[t] \rangle$, a positive integer $k$, and an online progression $\langle E_i[e_i], F_i[f_i] \rangle_j$, the incremental components are

$$\{B, P[t/1], P[t/2], \ldots, P[t/k], Q[t/k], E_1[e_1], E_2[e_2], \ldots, E_j[e_j], F_j[f_j]\}.$$

The $k$-expanded propositional formula $R_{j,k}$ of $\langle B, P[t], Q[t] \rangle$ w.r.t. $\langle E_i[e_i], F_i[f_i] \rangle_j$ is the conjunction of all these formulas.

# (Simple) Expansion vs. Incremental Composition

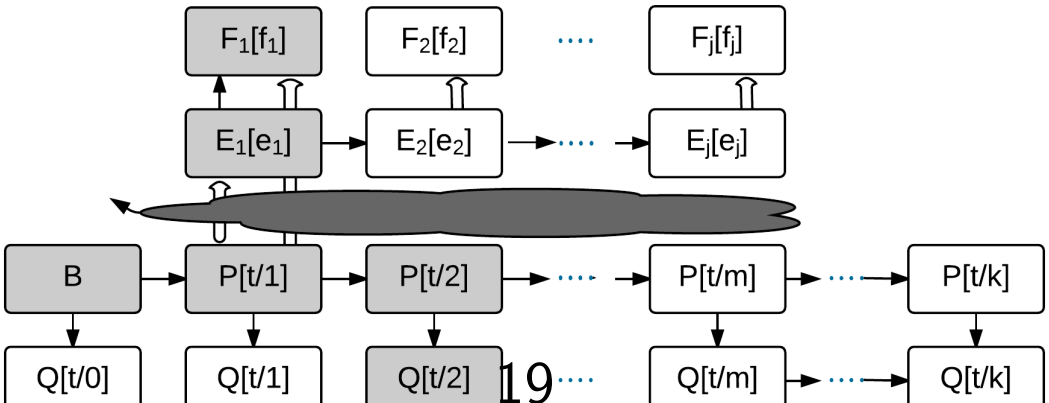
Given an incremental theory $\langle B, P[t], Q[t] \rangle$, a positive integer $k$, and an online progression $\langle E_i[e_i], F_i[f_i] \rangle_j$, the incremental components are

$$\{B, P[t/1], P[t/2], \ldots, P[t/k], Q[t/k], E_1[e_1], E_2[e_2], \ldots, E_j[e_j], F_j[f_j]\}.$$

The $k$-expanded propositional formula $R_{j,k}$ of $\langle B, P[t], Q[t] \rangle$ w.r.t. $\langle E_i[e_i], F_i[f_i] \rangle_j$ is the conjunction of all these formulas.

On the other hand, the incremental composition $\mathbb{R}_{j,k}$ "simplifies" each component first and "joins" incrementally, obeying some precedence order.

- Based on input/output stable models and the module theorem.
- Each component formula is associated with some designated set of external atoms, which are exempt from simplification.



Theorem 2 in the paper asserts that for an incremental theory and an online progression that are "modular" and "mutually revisable,"

$$R_{j,k} \text{ has the same stable models as } \mathbb{R}_{j,k}.$$

N.B. These conditions are highly complex and difficult to check manually. (See the paper.)

# $o\mathcal{BC}+$ into Incremental Theory

Given an $o\mathcal{BC}+$ action description $D$, an observation stream $\mathcal{O}$, and some formula $\mathcal{Q}[t]$, we define $\langle B, P[t], Q[t]\rangle^{D,\mathcal{Q}[t]}$ and $\langle E, F\rangle^{\mathcal{O}}$ as follows.

$$B = \bigwedge \begin{cases} 0:F \leftarrow 0:G & \text{for each static law in } D \\ 0:\{f=v\}^{\mathrm{ch}} & \text{for each regular fluent } f \text{ and each } v \in Dom(f) \\ 0:\{f=\mathtt{u}\}^{\mathrm{ch}} & \text{for each external fluent } f \\ 0:UEC_{\sigma^{fl}} & \end{cases}$$

$$P[t] = \bigwedge \begin{cases} t:F \leftarrow t:G & \text{for each static law in } D \\ (t-1):F \leftarrow (t-1):G & \text{for each action dynamic law in } D \\ t:F \leftarrow t:G \wedge (t-1):H & \text{for each fluent dynamic law in } D \\ t:\{f=\mathtt{u}\}^{\mathrm{ch}} & \text{for each external fluent } f \\ (t-1):\{a=\mathtt{u}\}^{\mathrm{ch}} & \text{for each external action } a \\ t:UEC_{\sigma^{fl}} & \\ (t-1):UEC_{\sigma^{act}} & \end{cases}$$

$$Q[t] = \neg\neg\mathcal{Q}[t]$$

$$E_i[m_i] = \bigwedge \begin{cases} m_i:c=v & \text{for each observation} \\ \neg\neg m_i:F & \text{for each observational constraint} \end{cases}$$

$$F_i[m_i] = \top$$

# Possible External Inputs

These atoms represent possible external inputs that may be introduced later by an online progression, and thus should be exempted from the current program simplification.

$At_{\mathrm{u}}(\sigma)$ denotes the set of all atoms $c=v$ such that $v \neq \mathrm{u}$.
We define the sets of explicit external inputs as follows:

- $I(B) = At_{\mathrm{u}}(0:\sigma^{ef})$,
- $I(P[t/i]) = At_{\mathrm{u}}(i:\sigma^{ef} \cup (i-1):\sigma^{ea})$,
- $I(Q[t/i]) = At_{\mathrm{u}}(\bigcup_{0\leq j<i}(j:\sigma^{ef} \cup j:\sigma^{ea}) \cup i:\sigma^{ef})$, and
- $I(E_i) = I(F_i) = \emptyset$.

# $o\mathcal{BC}+$ Ensures Modularity and Mutual Revisability

The translation of an $o\mathcal{BC}+$ description into propositional formulas ensures modularity and mutual revisability.

**Theorem (Modular and Mutually Revisable Construction)**

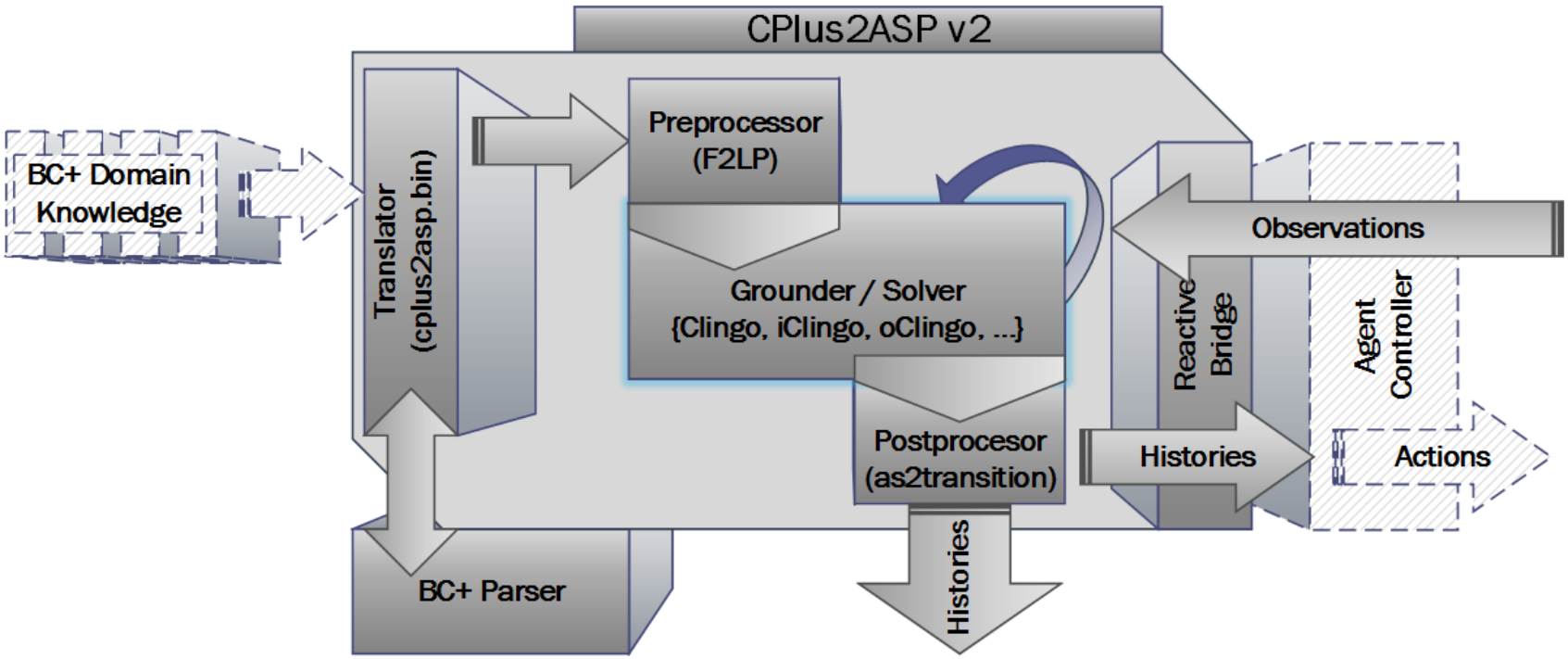
*Given an $o\mathcal{BC}+$ action description $D$ and an observation stream $\mathcal{O}$, and a step-parameterized formula $\mathcal{Q}[t]$, the corresponding incremental theory $\langle B, P[t], Q[t]\rangle^{D,\mathcal{Q}[t]}$ and the corresponding online progression $\langle E, F\rangle^{\mathcal{O}}$ are modular and mutually revisable.*

# Correctness of Incremental Composition

The stable models of the incremental composition represents the histories in the transition system that are normal with respect to the online stream.

## Theorem (Correctness of Incremental Composition)

*Given an o$\mathcal{BC}+$ action description $D$, an observation stream $\mathcal{O}$, a step-parameterized $\mathcal{Q}[t]$, and some $k \geq \widehat{m}$, let $\mathbb{R}_{\widehat{m},k} = \langle H, \mathcal{I}, \mathcal{O} \rangle$ be the incremental composition of $\langle B, P[t], Q[t] \rangle^{D,\mathcal{Q}[t]}$ w.r.t. $\langle E, F \rangle^{\mathcal{O}}$. The stable models of $H$ represents the histories of length $k$ in the transition system described by $D$ which (i) observe $\mathcal{O}$, (ii) are normal with respect to $\mathcal{O}$, and (iii) satisfy $\mathcal{Q}[t/k]$.*




The "reactive bridge" is a new software component, and acts as an intermediary between OCLINGO and a user-provided agent controller system. It allows the agent controller system to provide an $o\mathcal{BC}+$ observation stream during execution and receive updated solutions in the form of transition system histories.

# Experiments

In an elaboration of the Tower of Hanoi problem, an antagonistic entity moves pegs in order to attempt to thwart the agent's efforts.

```
 % abnormality
 ext_move(P,P1) causes ab_ext(P).
 ext_move(P,P1) causes ab_ext(P1).

% effect and precondition of moving
 loc(D)=L if -ab_ext(P) & -ab_ext(P1)
          after move(P,P1) & base(D)=P & clear(D)
                           & base(L)=P1 & clear(L).
 ext_move(P,P1) causes loc(D)=L if base(D)=P & clear(D)
                           & base(L)=P1 & clear(L).
```

"ab_ext(P)" is a regular fluent constant, which is asserted false by default.

In the presence of both agent's move(P,P1) and the antagonistic entity's ext_move(P,P1) leading to conflicting effects, ext_move overrides.

Given this program as input, CPLUS2ASP operates under the reactive mode listening to external observations. To find a solution for the original problem (assuming no external input), one may insert

```
#step.
#endstep.
```

CPLUS2ASP turns it into the input language of OCLINGO, runs OCLINGO and then again goes to the listening mode. One can then insert

```
#step.
observed ext_move(p2,p3) at 2.
#endstep.
```

and so on. In each observation of external actions, CPLUS2ASP does re-planning, but without grounding from scratch; only the new ground modules are added.

# Experiments

Highly effective in comparison with the offline execution, where grounding has to be done from scratch each time the external move is observed.

| | **No Ext. Input** | **1st Ext. Input** | **2nd Ext. Input** | **3rd Ext. Input** | **4th Ext. Input** | **Total Time**[a] |
|---|---|---|---|---|---|---|
| Online | 1.670[b] (0.320+1.350)[c] | 1.170 (0.360+0.810) | 0.420 (0.040+0.380) | 0.290 (0.070+0.220) | 0.270 (0.070+0.200) | **3.820** (0.860+2.960) |
| Offline | 3.840 (0.890+2.950) | 4.590 (0.890+3.700) | 4.430 (0.960+3.470) | 4.780 (1.110+3.670) | 4.920 (1.250+3.670) | **22.560** (5.100+17.460) |

[a] Accumulation of all iterations
[b] Total execution time (in seconds)
[c] grounding+solving

# Conclusion

- We extended the concept of online answer set solving to propositional formulas under the stable model semantics, and based on this, designed a high level online action language $o\mathcal{BC}+$, whose structure ensures the syntactic conditions that are required for the correctness of online answer set solving.
- We implemented the new language in CPLUS2ASP, which turns $o\mathcal{BC}+$ descriptions into the language of online answer set solver OCLINGO, and invokes OCLINGO for online computation.